LAB 6

MỤC TIÊU

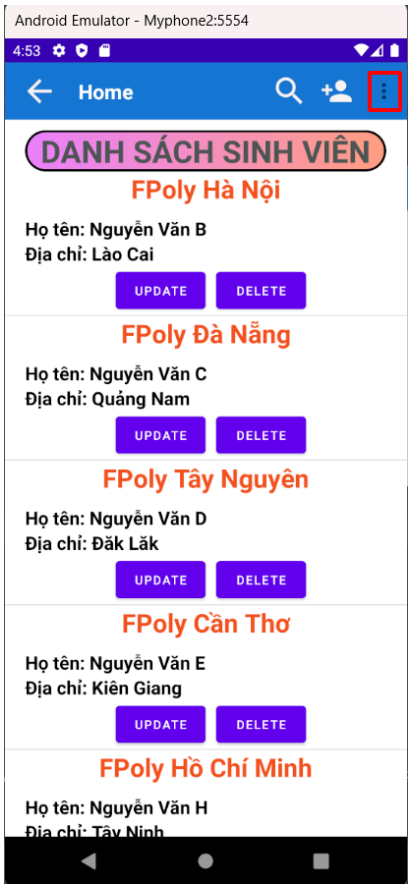
Kết thúc bài thực hành sinh viên có khả năng:

- ✓ Biết cách sử dụng menu, toolbar và custom toolbar.
- ✓ Biết cách sử tương tác với menu và toolbar.

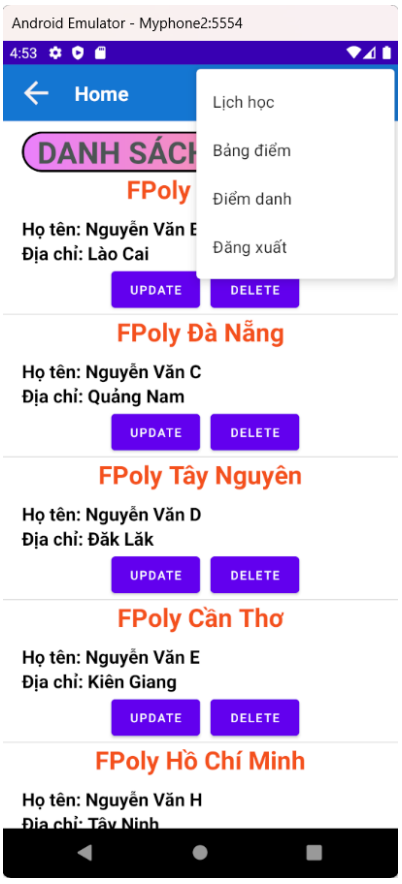
NỘI DUNG

BÀI 1:

Nội dung: Tiếp tục **Lab5** thiết kế giao diện như sau:



Trước khi click vào biểu tượng menu



Sau khi click vào biểu tượng menu

Hướng dẫn:

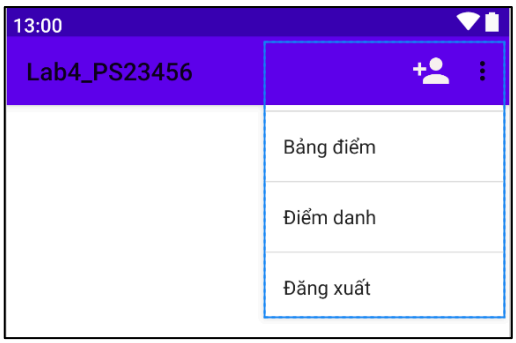
- Vào **res/values/themes/themes.xml** thay đổi parent thành **NoActionBar**:

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.Lab4_PS23456" parent="Theme.MaterialComponents.DayNight.NoActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_500</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
```

- Tạo file **toolbar.xml** trong thư mục **layout**, thiết kế giao diện như hình minh hoạ bên dưới (icon tuỳ chọn).
  Sử dụng thuộc tính **android:layout_height="?attr/actionBarSize"** để set chiều cao bằng với chiều cao của **Actionbar mặc định** (*khuyến nghị*).

```
<androidx.appcompat.widget.Toolbar
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/toolbar"
    android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
    android:background="#1976D2">
```

- Tạo menu cho ứng dụng với các chức năng sau:
  Thêm sinh viên mới, Bảng điểm, Điểm danh, Đăng xuất.
  Trong đó "Thêm sinh viên mới" set thuộc tính **app:showAsAction="always"**



- Trong layout **activity_home.xml** include toolbar vào giao diện:

```
<include layout="@layout/toolbar" />
```

- Ánh xạ và set toolbar:

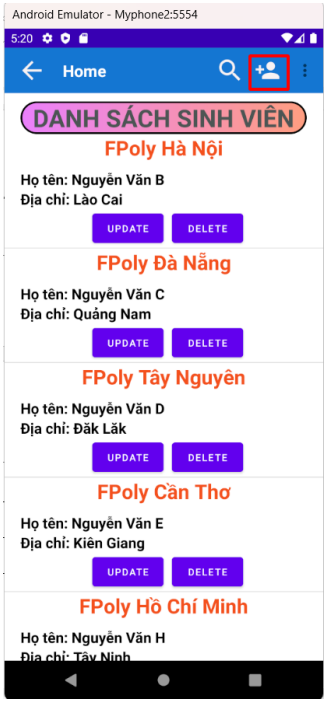
```
toolbar = findViewById(R.id.toolbar);
setSupportActionBar(toolbar);
```

- Khởi tạo menu:

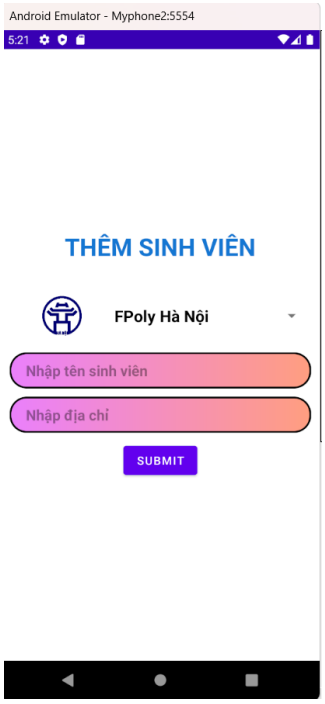
```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.menu, menu);
    return true;
}
```

BÀI 2: Tương tác với menu:

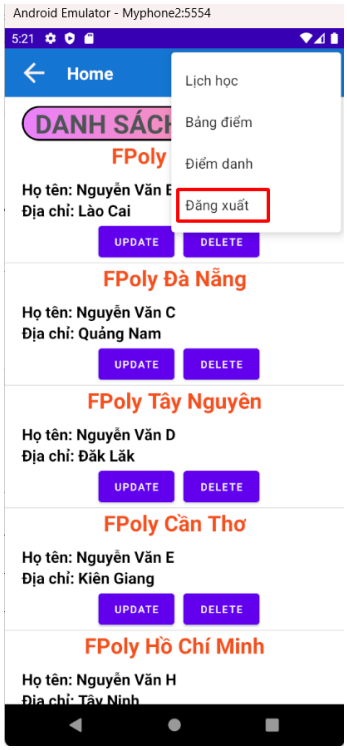
- ✓ Xây dựng chức năng thêm mới sinh viên bằng item menu.
- ✓ Xây dựng chức năng **Log Out.**
- ✓ Tham khảo minh hoạ dưới đây:



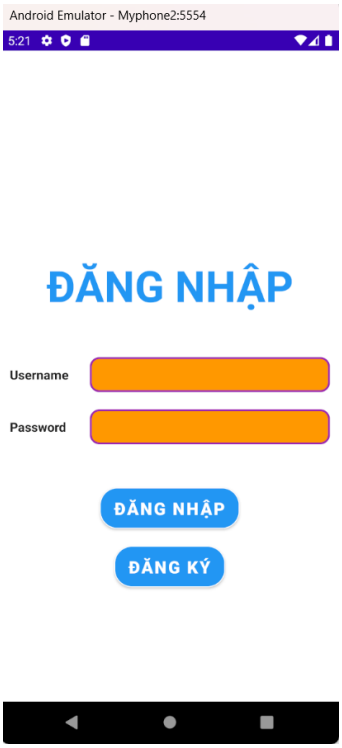
Trước khi click vào icon thêm sinh viên



Sau khi click vào icon thêm sinh viên



Trước khi click vào item đăng xuất



Sau khi click vào item đăng xuất

Hướng dẫn:

- Sử dụng hàm **onOptionsItemSelected** để bắt sự kiện khi click vào **item** trong **menu**:
  - ✓ Sử dụng code xử lý thêm mới sinh viên đã làm ở **Lab5** để xử lý sự kiện khi click vào **thêm sinh viên mới**.
  - ✓ Dùng Intent để thực hiện chức năng **Log Out**.
  - ✓ Với các item khác trong menu, Toast thông báo khi click vào item đó.

BÀI 3:

- Tiếp tục bài 2:
  - ✓ Xây dựng chức năng **tìm kiếm** sinh viên theo tên.
  - ✓ Xây dựng chức năng **quay lại** khi click vào **icon back** trên **toolbar**.

Hướng dẫn:

- Trong **AdapterListView** ta **implements Filterable** để thực hiện chức năng tìm kiếm:
  - ✓ Ta tạo thêm **List<Student> listOld**: dùng để lưu list ban đầu.
  - ✓ Đưa thêm trường này vào **constructor.**
  - ✓ Tham khảo code minh hoạ dưới đây:

```
public class ListStudentApdater extends BaseAdapter implements Filterable {

    11 usages
    private List<Student> list;
    4 usages
    private final List<Student> listOld;
    3 usages
    private final Context context;

    new *
    public ListStudentApdater(List<Student> list, Context context) {
        this.list = list;
        this.listOld = list;
        this.context = context;
    }
```

❖ **Implements** các hàm bắt buộc của class **Filterable**, xử lý sự kiện search tại đây:

```
@Override
public Filter getFilter() {
    Dinh Nguyen *
    return new Filter() {
        Dinh Nguyen *
        @Override
        protected FilterResults performFiltering(CharSequence charSequence) {
            String s = charSequence.toString();
            if (s.isEmpty()) {
                list = listOld;
            } else {
                List<Student> listS = new ArrayList<>();
                for (Student st : listOld) {
                    if (st.getName().toLowerCase().contains(s.toLowerCase())) {
                        listS.add(st);
                    }
                }
                list = listS;
            }
            FilterResults filterResults = new FilterResults();
            filterResults.values = list;
            return filterResults;
        }

        Dinh Nguyen
        @Override
        protected void publishResults(CharSequence charSequence, FilterResults filterResults) {
            list = (List<Student>) filterResults.values;
            notifyDataSetChanged();
        }
    };
}
```

❖ Trong activity **Home,** bắt sự kiện cho **EditText:**

```
edtSearch.addTextChangedListener(new TextWatcher() {
    1 usage  Dinh Nguyen
    @Override
    public void beforeTextChanged(CharSequence charSequence, int i, int i1, int i2) {

    }

    Dinh Nguyen *
    @Override
    public void onTextChanged(CharSequence charSequence, int i, int i1, int i2) {
        listStudentApdater.getFilter().filter(charSequence);
    }

    Dinh Nguyen
    @Override
    public void afterTextChanged(Editable editable) {

    }
});
```

❖ Đối với chức năng **quay lại**, sinh viên tự thực hiện.

BÀI 4: GV CHO THÊM

*** YÊU CẦU NỘP BÀI:

Sv nén file bao gồm các yêu cầu đã thực hiện trên, nộp lms đúng thời gian quy định của giảng viên. Không nộp bài coi như không có điểm.

--- Hết ---